Invited Review: A Guide on How to Avoid Common Pitfalls in Model Performance Metrics and Validation

Chun-Peng James Chen

# Introduction

Modeling is an indispensable tool for hypothesis formulation and decision-making. It functions as a structured framework that validates system understanding through the analysis of empirical data, and further extends this understanding by enabling the extrapolation of results to novel trials and conditions. The advancement of research is fundamentally built upon the accumulation of prior knowledge within the scientific community. Therefore, the evaluation of model performance becomes particularly critical, necessitating a rigorous and standardized approach that allows for both reproducibility and comparability. The failure to adhere to these standards, by reporting model performance through ill-defined metrics or non-rigorous procedures, has the potential to introduce misinterpretations and miscommunications. Such lapses not only impede scientific progress but can also compromise the integrity of the collective body of research in the field.

This review aims to scrutinize four common pitfalls in model evaluation. The first pitfall is the failure to validate model performance on unseen or test datasets; the second involves indiscriminate feature selection across the full dataset; the third emerges when tuning hyperparameters with the same data designated for model evaluation; the fourth is neglecting block effects in model validation. In each section, current relevant practices in dairy science are examined to offer practical examples for clearer illustration. To underscore the importance of these pitfalls, a minimally reproducible simulation is presented to demonstrate their impact on model evaluation.

This review tried to answer commonly asked questions. For example, how to choose between leave-one-out cross validation and five-fold cross validation? How many samples are enough to unbiasedly estimate a model performance? There is no single approach can apply to all research problem, this review aims to provide a comprehensive perspective of evaluating models with different types of metrics and validation strategy.

# Performance Metric

Performance metrics serve as quantitative indicators for evaluating model performance. They are essential tools for benchmarking various modeling approaches and validating hypotheses. However, it's important to note that the choice of metrics can significantly influence the evaluation results. Depending on the specific hypothesis being tested, an overly optimistic assessment may be concluded if inappropriate metrics are selected. This section aims to introduce commonly used performance metrics in the field of dairy science and discuss potential pitfalls that researchers should be cautious of.

## Regression

A regression model aims to predict a continuous variable and is commonly employed in various applications, such as estimating milk composition, yield, and feed efficiency, as well as assessing environmental impacts in livestock production. This section delves into three widely used metrics for evaluating regression models: Pearson's Correlation Coefficient ($r$), Root Mean Squared Error (RMSE), and the Coefficient of Determination ($R^2$).

A group of graphs with numbers and points

Description automatically generated with medium confidence

In the hypothetical example depicted in Figure 1, 100 observations are generated from two separate normal distributions. The first 50 observations are drawn from a normal distribution with a mean of -3 and a standard deviation of 1, denoted as \( \mathcal{N}(-3, 1) \). The remaining 50 observations are generated from another normal distribution, \( \mathcal{N}(3, 1) \). Utilizing two distinct distributions serves to simulate experimental block effects, preset at a magnitude of 6 units for this experiment.

Based on these simulated observations, four sets of predictions are generated:

* **First Prediction**: Each observation is multiplied by 0.3 to establish a correlation, followed by the addition of random noise \( \mathcal{N}(0, 0.7) \) to introduce prediction errors.
* **Second Prediction**: Values from the first prediction are multiplied by 5, simulating predictions with a larger variance while maintaining the same relative ordering with the original predictions.
* **Third Prediction**: Each value from the first prediction is raised to the power of 5. This transformation serves to compress values within the range of -1 to 1 towards zero, without affecting their relative order. Additionally, values greater than 1 or less than -1 are pushed farther from zero, simulating outlier predictions.
* **Fourth Prediction**: Values sampled from \( \mathcal{N}(-3, 1) \) and \( \mathcal{N}(3, 1) \) are added to the first and second blocks of observations, respectively. This amplifies the block effects, simulating a model that effectively distinguishes between different blocks but is less capable of predicting individual variations within each block.

This quartet of predictions serves to simulate potential challenges and complexities encountered in real-world modeling scenarios, thereby providing a foundation for evaluating different performance metrics used in regression problems.

Explain the intension of each case

### Coefficient of Determination ()

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

The Coefficient of Determination, commonly known as R-squared \( R^2 \), provides a similar measurement of the SSE as RMSE, but it is normalized by the total sum of squares (SST) to yield a more versatile metric for comparing results across different experiments. As defined in Equation \(3\), \( n \) represents the number of observations, \( y \) signifies the actual observed values, \( \hat{y} \) indicates the predicted values, and \( \bar{y} \) is the mean of the observed values. A higher \( R^2 \) value suggests a lower prediction error. Importantly, if \( R^2 \) falls below zero, the predictions are deemed inferior to a naive approach that predicts the mean of the observed values for all samples, particularly in terms of absolute prediction errors. Similar to RMSE, \( R^2 \) is influenced by the variance of the predictions. Specifically, when predictions exhibit a high degree of variance that deviates from the actual observations, the \( R^2 \) value can be adversely impacted, dropping from 0.47 in Prediction 1 to -0.27 in Prediction 2, for example. Additionally, due to the presence of squared terms in its calculation, \( R^2 \) is sensitive to outliers, as evidenced in Prediction 3.

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

The fourth prediction scenario offers an intriguing case study for \( R^2 \) to identify misleading model performance due to block effects. According to Equations \(4\) and \(5\), the SST can be decomposed into the Regression Sum of Squares (SSR), which represents the variation explained by the model, and the SSE, which accounts for the unexplainable errors as discussed in the RMSE section. From this standpoint, \( R^2 \) quantifies the proportion of variation that the model is able to explain. In the case of the fourth prediction, the model effectively captures the differences between various blocks, which contributes substantially to the SST. Consequently, the \( R^2 \) value rises to as high as 0.80. However, this elevated metric can be misleading. The hypothetical model in the fourth prediction does not possess the ability to capture individual variations within each block. The strength of \( R^2 \) lies in its capability to differentiate between sources of variation. Upon closer inspection of the data within each block, the \( R^2 \) values plummet to -0.71 and -1.10, respectively, highlighting that the model fails to account for intra-block variability. In summary, while both RMSE and \( R^2 \) aim to measure prediction errors, \( R^2 \) offers additional statistical insights that facilitate a more nuanced evaluation of model performance.

Current practices in dairy science usually report both R2 and RMSE, and sometimes r correlation. as exemplified in <example *1*>, <example *2*>.

### Pearson Correlation Coefficient (r)

|  |  |  |
| --- | --- | --- |
|  |  | () |

The Pearson Correlation Coefficient (\(r\)) is a widely used metric for assessing the linear relationship between two variables, as defined by Equation \(1\). In this equation, the observed values are denoted by \(x\) and the predicted values are represented by \(y\). To calculate the correlation coefficient, the equation first computes the covariance between \(x\) and \(y\), which captures how the coordinates of data points deviate from their means. This value is then normalized by the product of the standard deviations of \(x\) and \(y\). The coefficient can be either positive or negative, reflecting a positive or negative correlation between the two variables, respectively. Owing to the normalization term in the denominator, the coefficient is scale-invariant and will always fall within the range of -1 to 1. This attribute is illustrated in the first two scenarios: despite the second scenario having a variance five times larger than the first, the coefficient remains consistent.

However, the metric has its limitations. It is sensitive to the presence of outliers, as shown in the third scenario where most data points cluster near zero but a few outliers yield a high correlation coefficient. Similarly, the metric is influenced by block effects, leading to inflated correlation values, as observed in the fourth scenario. When calculated within individual blocks, the coefficients drop to 0.11 and 0.06, respectively. This emphasizes the importance of either visually inspecting regression results through scatter plots or examining them within individual blocks. If the objective is to examine the model predictability of individual variation rather than block effects, caution should be exercised when applying this metric, especially in scenarios where block effects are evident.

In practice, this coefficient is often utilized to evaluate a model capability to correctly order or rank individual observations based on a specific trait of interest. For instance, <example *1*>, <example *2*>, among others [ref 3-6].

### Root Mean Squared Error (RMSE)

|  |  |  |
| --- | --- | --- |
|  |  | () |
|  |  |  |

The Root Mean Squared Error (RMSE) serves as a quantitative measure to gauge the average magnitude of prediction errors between observed values (\(y\)) and their predicted values (\(\hat{y}\)). It gives the error in the same units as the observation (\(y\)), and a lower RMSE value indicates a better model performance. Defined by Equation \(2\), \(n\) stands for the number of observations. Distinct from the correlation coefficient, RMSE is sensitive to scale, implying that achieving predictions with a variance akin to the observed values takes precedence over maintaining their order or trend. This is particularly pertinent when the focus is on the absolute magnitude of the error. Take for instance Scenario 2, where the predictions have been scaled by a factor of 5 compared to Scenario 1. The RMSE inflates from 2.41 to 3.63, underscoring that even if both scenarios rank the observations identically, RMSE effectively captures the discrepancies in the absolute errors. Another notable characteristic of RMSE is its sensitivity to outliers. In Prediction 3, where certain predictions deviate substantially from the majority, the squaring operation within RMSE accentuates these outliers, culminating in a RMSE value of 25.56. It's also worth mentioning that RMSE is impervious to block effects, unlike the correlation coefficient. In Prediction 4, both the complete set of predictions and the intra-block predictions yield comparable RMSE values—1.49, 1.46, and 1.52, respectively.

In case 4 RMSE decreased which is not an good indicator

Due to its emphasis on absolute error, RMSE is frequently employed in contexts where the magnitude of the error is a critical consideration. For example, <example *1*> and <example *2*>.

## Classification

Classification models aim to predict categorical outcomes such as 'healthy' or 'sick', 'susceptible' or 'resistant', and 'high yield' or 'low yield'. This section presents a hypothetical example to highlight how the choice of performance metrics can lead to misleading model evaluations. Consider a binary classification model predicted on ten samples, of which four are positive (+) and six are negative (-), and the model produces a list of probabilities between 0 and 1, indicating the likelihood that each sample belongs to the positive class. The prediction can be expressed as the following joint probability distribution:

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

Where $X$ is a random variable representing the predicted probabilities, and $Y$ representing the ground truth labels. denotes a uniform distribution between $a$ and $b$. Table 1 shows ten samples drawn from this distribution to simulate the model predictions.

A table with numbers and labels

Description automatically generated

This example illustrates a scenario where the positive outcome is rare, which is commonly encountered in abnormality detection tasks. Before judging the prediction outcomes, a threshold that defines how the prediction probability is converted to a binary outcome is needed. If a prediction probability is greater than the threshold, the sample is classified as positive. In this example, the threshold is set to 0.5 for simplicity, the following confusion matrix summarizes the model performance (Figure 2a). Noted that the threshold can be adjusted for specific applications and change the confusion matrix accordingly.

This illustrate the variant outcomes from the same prediction, when the metric is different.

Assigning the positive to the rare category can result in different conclusion

A screenshot of a computer

Description automatically generated

### Accuracy

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

Accuracy is a straightforward metric in classification problems, as defined in Equation 6. Here, TP, TN, FP, and FN represent true positive, true negative, false positive, and false negative, respectively. Based on this definition (Equation 6) and the confusion matrix, the model performance from the example is 0.60, which may appear to be better than random guesses (0.50). However, this metric could be misleading when applied to imbalanced datasets, warranting the use of additional metrics for a comprehensive evaluation. However, caution must be exercised when applying this metric to imbalanced datasets. In such cases, a model could show an accuracy that is higher than random guessing by predicting all samples as negative in an imbalanced dataset where negatives are predominant. This demonstrates that solely relying on accuracy is insufficient for evaluating a classification model, particularly when dealing with imbalanced datasets. Therefore, it is crucial to consider additional metrics for a more comprehensive and robust evaluation.

### Precision, Recall, and Precision-Recall Curve

|  |  |  |
| --- | --- | --- |
|  |  | (7)  (8) |

Precision evaluates the proportion of true positive predictions among all positive predictions. It effectively measures how reliable a positive prediction is. Recall, or sensitivity, measures the proportion of true positives among all actual positives. It gauges how effectively the model identifies positive samples. For instance, if the threshold is set as low as 0.1, the model is prone to making false positives, resulting in low precision. A high rate of false positives could be particularly costly in applications like <example *1*>, where unnecessary treatments could be administered based on these incorrect results. Conversely, the same low threshold can yield high recall as the model is less likely to miss actual positives. In situations where failing to identify a positive instance can have severe consequences, such as <example *2*> failing to detect a disease in its early stages, high recall could be more valuable.

A screenshot of a graph

Description automatically generated

When applying the metrics of precision and recall to the hypothetical example, the precision and recall values are 0.5 and 0.25, respectively. Both metrics yield lower values than the calculated accuracy of 0.6, underscoring the necessity of utilizing multiple metrics for a thorough evaluation of model performance. However, it is worth noting that both metrics are primarily focused on the classification of positive samples. This can introduce bias and can be particularly problematic when dealing with imbalanced datasets. As an example, if there is a scenario where the sample labels get reversed (i.e. the positives become negatives and vice versa) but the model parameters remain unchanged, the precision and recall values will shift to 0.625 and 0.833, respectively, as shown in Figure 2b. This outcome suggests that relying solely on precision and recall may not provide a complete understanding of model performance in certain contexts. Although incorporating metrics that focus on negative samples, such as specificity, can partially mitigate this issue, there remains a clear need for more robust and label-invariant metrics for an unbiased and comprehensive evaluation.

The trade-off between precision and recall is presented by adjusting the threshold. As the threshold increases, the model becomes more conservative and predicts fewer positive samples, yielding higher precision and lower recall. Since the precision and recall values reported from one single confusion matrix can only represent one specific threshold, the precision-recall (PR) curve is introduced to provide a more comprehensive view of the model performance. In the PR curve, the x-axis represents the recall values and the y-axis represents the precision values. The curve is derived by calculating the precision and recall values at different thresholds (Figure 2b). The area under the curve (AUC) is a common metric to summarize the performance of the PR curve. Still, the AUC is label-dependent, showing 0.76 for the original labels and 0.94 for the reversed labels.

Serve as a visual tool to optimize the threshold if one seek for a higher recall with the constant precision level.

### Receiver Operating Characteristic (ROC) Curve

The Receiver Operating Characteristic (ROC) curve represents the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. It is a label-invariant and threshold-invariant alternative to the PR curve. The x-axis is the FPR and the y-axis is the TPR, which is equivalent to recall.

|  |  |  |
| --- | --- | --- |
|  |  | (7)  (8) |

The ROC curve can be interpreted as how much cost is needed to capture true positives. If the curve climbs steeply from the left side, it means that the model can capture most true positives with a low cost of false positives. A random guess, which yield a 50% chance of making a correct prediction, is represented by a diagnoal line in the ROC curve. This curve is widely used in reporting genetics markers in Genome-Wide Association Studies (GWAS) [ref 7-9], as whether the top-associated markers (i.e., prediction with high positive probability) can be identified is more important than considering the entire list of prediction quality. In the hypothetical example, the ROC curve show a labe-invariant AUC of 0.875, which is the same for both the original labels and the reversed labels. However, such high metric may still mislead the conclusion of model evalution, as it does not reflect the poor quality that the model has in predicting positive samples.

At the same level, when 0.25, inverted is worse, where 0.50, inverted get a better result.

### Matthews Correlation Coefficient (MCC)

The Matthews Correlation Coefficient (MCC) has been proposed as a more robust metric for the evaluation of binary classification models, particularly in the context of imbalanced datasets [refs 10, 11]. The MCC is defined as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (11) |

This metric comprehensively incorporates all four elements of the confusion matrix, rendering it invariant to changes in label assignments. The MCC ranges from -1 to 1, with 1 denoting perfect classification devoid of false positives and false negatives, and 0 suggesting performance equivalent to random guessing. In a hypothetical case study using a threshold of 0.5, the MCC value was found to be 0.10, signifying suboptimal classification quality. This evaluation aligns more closely with the expected model performance compared to metrics like accuracy, ROC AUC, or precision and recall, particularly when the dataset skews toward positive samples. The characteristics of MCC make it a valuable tool for identifying optimized classification thresholds. By evaluating the MCC across different thresholds, one can pinpoint the threshold that maximizes the MCC value, thereby enhancing the model's overall performance. For instance, in the aforementioned example, the MCC reached its maximum value of 0.82 at a threshold of 0.2. This threshold yielded accuracy, precision, and recall values of 0.90, 0.75, and 1.00, respectively. Interestingly, even when labels were swapped, the highest MCC value remained the same, with values of 0.90 for accuracy, 1.00 for precision, and 0.83 for recall. This serves to underscore the MCC's balanced consideration of both positive and negative samples, solidifying its role as a versatile metric for a well-rounded model evaluation.

A graph with green and orange lines

Description automatically generated

It should carefully consider which label to be positive. Usually, if the event is of interest, it should be prioritized to be the positive label as both precision and recall focus on the predictability of positive samples.

Although it is rare to be visualized as a curve, it is a convenienct way as a single indicator to find the optimum threshold that achieve high precision and recall.

# Model Validation

Model validation aims to evaluate how well a given model generalizes to an independent dataset that it has not seen during the training process. The most common methods for model validation are K-fold cross-validation (K-fold CV). To implement the K-fold CV, the available dataset, denoted as , is partitioned into equally sized folds. Mathematically, we can represent the dataset as:

|  |  |  |
| --- | --- | --- |
|  |  | (12) |
|  |  |  |

where \( X \in \mathbb{R}^{n \times p} \) represents the input features, and \( Y \in \mathbb{R}^{n \times 1} \) symbolizes the ground truth labels. \( n \) is the total number of samples, while \( p \) indicates the number of features. In each iteration of the K-fold CV, one fold is reserved as the test set, (or ), to act as unseen data, while the remaining folds make up the training set (or ):

|  |  |  |
| --- | --- | --- |
|  |  | (13)  (14) |

After splitting the dataset into and , the examined model is trained on the training set and denoted as. The hold-out test set is then used to evaluate the model performance , which is defined by comparing the predicted labels with the true labels using a performance metric , such as the root mean squared error (RMSE) or determination coefficient :

|  |  |  |
| --- | --- | --- |
|  |  | (15) |

The K-fold CV procedure repeats the above process times until each fold has been used as the test set once. Lastly, the estimated expected generalization performance of the model leveraging the entire dataset is the average of the prediction performance over the all folds:

|  |  |  |
| --- | --- | --- |
|  |  | (15) |

## Validation Bias and Variance

### Definition

The true generalization performance of the model $G(f\_{\mathcal{D}})$ can only be approximated by averaging the performance metrics over infinite unseen datasets. However, in practice, the dataset $\mathcal{D}$ is finite and therefore there is always a bias in the estimation of $G(f\_{\mathcal{D}})$. The validation bias can be defined as:

|  |  |  |
| --- | --- | --- |
|  |  | (15) |

For example, if RMSE is used as the performance metric, a positive validation bias suggests that the model validation procedure concludes a pessimistic estimation of the model performance, since the true performance is expected to be lower than the estimated performance. Another aspect of model validation is the variance of the estimated performance. For example, in a 5-fold cross validation, there are five estimates of the model performance. The variance among these five estimates is the validation variance. A high validation variance suggests that the performance is sensitive to the choice of the test set $\mathcal{D}\_{\text{k}}$, which may be caused by small sample size or over-complex model. The validation variance can be defined as:

|  |  |  |
| --- | --- | --- |
|  |  | (15) |

It is noted that $\mathbb{E}[\hat{g}(f\_{\mathcal{D}\_\text{-k}})]$ is equivalent to $\mathbb{E}[\hat{g}(f\_{\mathcal{D}})]$ in K-fold CV, since $\mathbb{E}[\hat{g}(f\_{\mathcal{D}})]$ is estimated by averaging all $\hat{g}(f\_{\mathcal{D}\_\text{-k}})$ over $K$ folds, which is the definition of $\mathbb{E}[\hat{g}(f\_{\mathcal{D}\_\text{-k}})]$. Combining the bias and variance, the mean squared error (MSE) of the model validation can be defined as:

|  |  |  |
| --- | --- | --- |
|  |  | (15) |

### Bias-Variance Trade-off

From the equpation above, a trade-off relationship between the bias and variance can be observed given a constant validation MSE. With the fixed sample size and model complexity in K-fold CV, the choice of $K$ is the major factor that affects the bias and variance of the model validation. When the $K$ is set larger, each training set $\mathcal{D}\_{\text{-k}}$ is larger in size, whcih means the model is trained on a dataset that is more representative of the population of interest, leading to lower bias. However, because the test set $\mathcal{D}\_{\text{k}}$ is relatively small in size, the validation variance can be high due to the high sensitivity to the specific data points in the test set $\mathcal{D}\_{\text{k}}$. On the other hand, with fewer folds when $K$ is set smaller, each training set $\mathcal{D}\_{\text{-k}}$ is smaller, leading to worse representation of the population and higher bias. However, the test set $\mathcal{D}\_{\text{k}}$ is larger in size, in which the estimate from each fold is more stable and therefore the validation variance is lower.

Leave-one-out cross validation (LOOCV) is a special case of K-fold CV where $K$ is equal to sample size $\mathcal{N}$ in the complete dataset $\mathcal{D}$. It is known for an unbiased estimation of the model performance, since the training set $\mathcal{D}\_{\text{-k}}$ has good representative sample size $\mathcal{N} - 1$. However, as the trade-off discussion suggested, the validation variance can be also high as there is only one sample being tested in each fold. It is worth noting that the unbiased estimation in LOOCV can only be achieved when all $K$ folds are tested. If an incomplete LOOCV is conducted, the validation bias can also be high due to its nature of high validation variance. Therefore, LOOCV should be avoided when the dataset is large and the training process is costly in time or computational resources. A detailed discussion of the trade-off has been discussed in (ref1, ref2).

### Simulation Objectives and Hypothesis

A simulation study is conducted to examine the interaction between sample sizes and various performance estimators, as well as how this interaction influences the bias and variance in model validation. It is hypothesized that both bias and variance will diminish as the sample size grows. Furthermore, it is anticipated that the variance will escalate with an increase in the number of folds used by the estimator, although this will concurrently decrease bias. Given that K-fold cross-validation (CV) utilizes only a portion (i.e., \( K - 1 \) folds) of data points for training, it is considered a pessimistic estimate of model performance. The study also aims to quantify the extent of performance underestimation for each CV estimator.

### Simulation Design

The studied performance estimators include K-fold cross validation where K is assigned as 2, 5, and 10, and LOOCV, which is a special case of K-fold CV where K is equal to the sample size, and "In-Sample" that validates the model on the same dataset used for training. The "In-Sample" metric is presented to demonstrate an overoptimistic estimation of the model performance without conducting model validation. Three performance metrics are used to evaluate the model performance, including the RMSE (eq x), $R^2$ (eq y), and $r$ (eq z). The validating model in this simulation is multivariate linear regression which takes ten features as the input regressors and one target variable as the output. Both the input regressors and target variable are generated from a normal distribution with mean 0 and standard deviation 1. Hence there should have no linear association being observed from the data. The sample size $\text{n}$ is set as 50, 100, and 500 to observe the interaction between the sample size and the performance estimators. The simulation was iterated 1000 times for each setting (i.e., sample size and estimator) to observe the distribution of bias and variance.

In each iteration of the simulation, the dataset $\mathcal{D}=\{X, Y\}$ is sampled accordingly to the simulation assumption as described. If the estimator is a K-fold CV, the dataset $\mathcal{D}$ is partitioned into $K$ folds in which each fold is $\mathcal{D\_k}=\{X\_k, Y\_k\}$. Otherwise, the dataset $\mathcal{D}$ is not partitioned in the "In-Sample" estimator. The linear model $f$ is trained on the training set $\mathcal{D}\_{\text{-k}}$ to estimate the coefficients $\beta$, which is then used to predict the target variable $\hat{Y\_k}$ in the test set $\mathcal{D}\_{\text{k}}$. The procedure of K-fold CV can be expressed as:

|  |  |  |
| --- | --- | --- |
|  |  | (20) |

Since there is no split in the "In-Sample" estimator, the prediction of the target variable $\hat{Y}$ in "In-Sample" is obtained as:

|  |  |  |
| --- | --- | --- |
|  |  | (20) |

where:

- $X$ is the input regressors sampled from a standard normal distribution $\mathcal{N}(0, 1)$ with dimensions $\text{n} \times 10$.

- $Y$ is the target variable sampled from a standard normal distribution $\mathcal{N}(0, 1)$ and belongs to $\mathbb{R}^{\text{n} \times 1}$.

- $X\_{\text{-k}}$ and $Y\_{\text{-k}}$ are the input regressors and target variable in the training set $\mathcal{D}\_{\text{-k}}$.

- $X\_{\text{k}}$ is the input regressors in the test set $\mathcal{D}\_{\text{k}}$.

- $\hat{Y}\_{\text{k}}$ is the predicted target variable in the test set $\mathcal{D}\_{\text{k}}$.

- $\beta$ is the estimated regression coefficients and belongs to $\mathbb{R}^{10 \times 1}$.

- $\epsilon$ is the error term assumed to be normally distributed with mean 0 and standard deviation 1.

After obtaining the estimated target variable $\hat{Y\_k}$ or $\hat{Y}$, the estimated performance $\mathbb{E}[\hat{g}(f\_{\mathcal{D}})]$ can be derived as described in the previous section. To simulate the true model performance $G(f\_{\mathcal{D}})$, one-hundread of unseen datasets $\mathcal{D}^{\*}$ are sampled in the same manner as the dataset $\mathcal{D}$, and the model performance $G(f\_{\mathcal{D}})$ is approximated by averaging the performance metrics over all $\mathcal{D}^{\*}$. The bias and variance of the model validation are then calculated as eq x and eq y, respectively.

### Results

A diagram of a graph

Description automatically generated with medium confidence

The 100-iteration simulation results were summarized in the box plots to examine the validation bias and variance distribution. The figure 1 focuses on examining the bias changes across different estimators and sample sizes. Regardless of estimator and metric, the bias decreases as the sample size escalated. Although LOOCV has been considered as the

A screenshot of a graph

Description automatically generated

When the metrics are $R^2$ or RMSE, LOOCV

The to observe the trade-off relatinoship

Considering that there is only one data point tested in LOOCV, the validation variance is only applicable to the metric RMSE. Figure 1.1 inllustrates the bias and variance in the RMSE across different performanc estimators as a function of sample size $\text{N}$. Both the bias and variance in RMSE are observed to decrease as the sample size increases, which meet the hypothesis. The LOOCV is found to have the least biased estimation among all the estimators. Although 2-fold CV shows the highest bias, however, the bias did not show a significant decrease when the sample size increases. And all estimator shows similar bias when the sample size reaches 500. Rregarding validation variance, LOOCV exhibits a consistently higher value as compared to other estimators across all sample sizes. Furthermore, it is observed that a lower number of folds $K$ correlates with reduced variance, which is also consistent with the hypothesis.

When including other metrics suchs as $r$ and $R^2$ (figure 1.2), and In-Sample approach into the discussion, the

### Conclusion

## Model Selection

### Hyperparameter tuning and feature selection

Model selection is required when the model are not solely determined by the data. For example, a regularized linear regression model, such as ridge regression or least absolute shrinkage and selection operator (LASSO), has to define a regularization term λ before fitting the model to the given data. A larger λ will result in a more regularized model, which tends to shrink small coefficients to zero to avoid overfitting to noise in the training data. Below are the loss functions between unregularized OLS and regularized ridge regression and LASSO regression:

|  |  |  |
| --- | --- | --- |
|  |  | (20) |

Where $x\_i$ and $y\_i$ are the $i^{th}$ row of the design matrix $X$ and the response vector $y$, respectively. And $n$ is the sample size and $\beta$ is the coefficient vector. As all three model aims to find the optimal β that minimizes the loss function $\mathcal{L}$, the length of $\beta$ in the regularized models (i.e., ridge and LASSO regression) is penalized in the loss function.

These parameters that defined how the model is fitted and are not changed during the training process are called hyperparameters. In addition to the regularized models, hyperparameters are widely used in other prediction models for better flexibility and robustness. For example, a support vector regression (SVR)projects the regressors ($X$) onto a linear subspace to approximate the target variable $y$. However, the incorporation of a kernel function permits the SVR to further explain non-linear relationships within the data. Picking an appropriate hyperparameter, kernel function, to best fit the data can enhance the model performance in non-linear data. Another hyperparameter example is the size of the latent variables in partial least square regression (PLSR), which compresses the original regressors into a smaller set of latent variables to avoid multicollinearity problems. A lower number of latent variables will lose more information from the original regressors, while a higher number of latent variables will result in overfitting. Selecting the optimal value for these hyperparameters is known as model selection (Himeldorf and Wahba) or hyperparameter tuning.

In addition to hyperparameters, feature selection is another type of model selection in which the model is fitted to a subset of the original regressors. This procedure is commonly required in dealing with high-dimensional data, where the number of features or regressors are much larger than the number of observations and resulting poor generalization performance. For example <list spectral study and GWAS>.

When implementing the model selection, a common pitfall is to exclude the selection from the model validation process. For example, when studying production traits using hyperspectral devices where hundreds of spectral bands are available, determining the ideal subset of the bands and model hyperparameters are the essential step before start training the model. The risk of overestimating the model performance emerges when the optimal spectral bands are selected based on the performance on the test set. Even the selected model will undergo a k-fold cross validation, the model has been selected in favor of the test set therefore overestimate the model performance. This validation mistake has been discussed in many literature <list literature> and should be carefully avoided in practice. A workaround is to further split the training/test sets to training/validation/test sets, where the validation set is used to select the optimal model and leave the test set untouched throughout the training process.

Decision tree also involve hyperparameters such as tree depth and

### Simulation Objectives and Hypothesis

This simulation study is to investigate the impact of falsely impelemting model selection on the validation bias. The examined model selection procedures includes feature selections and hyperparameter tuning. The hypothesis is that the model performance will be significantly overestimated when the test set is incorrectly used in either of the model selection procedures.

### Simulation Design

### A diagram of a program Description automatically generated

A regression task is simulated in this study. Support Vector Regressor is used as the model, which applies a kernel function to project a subset of features $X$ to predict the label $y$. The features $X$ and label $y$ variables are sampled from a normal distribution, which provides a baseline correlation performance $r=0$ for estimating the validation bias. The sample and feature sizes are set to 100 and 1000, respectively. The feature selection process is conducted by selection the top 50 features with the highest correlation with the label $y$. In tuning the hyperparameters, the kernel functions: linear, polynomial, radial basis function (RBF), and sigmoid functions, were evaluated in the tuning process.

Notations $FS$ and $HT$ were used to denote feature selection and hyperparameter tuning, respectively. And a binary value 0 or 1 to indicate whether the model selection is implemented correctly, where 1 indicates a correct implementation. With this setting, there are a total of four different model selection combinations: 1. $FS=0;HT=0$, 2. $FS=0;HT=1$, 3. $FS=1;HT=0$, and 4. $FS=1;HT=1$. When $FS=0$, the features are selected before the data is split for the cross validation, otherwise the features are selected within each cross validation fold using only the training set. For the hyperparameter turning, the dataset will be split to training, validation, and test sets when $HT=1$. The model will be trained on the training set and evaluated the performance on the validation set for each hyperparameter. The test set will only be used once in reporting the estimated performance of the selected model. On the other hand, when $HT=0$, the data will only be split into training and test set. The model will be trained on the training set and evaluated the performance on the test set for each hyperparameter. The test set will be used multiple times, and only the best performance among all hyperparameters will be reported. The number of folds is set to 5. For example, when $HT=1$, the dataset will be first allocated 80% to training set, and 20% to the test set. Then, the training set will be further split into five folds, of which four folds of them are training set (64% of the entire dataset) and one fold is used as the validation set (16% of the entire dataset).

The validation bias is estimated by the metric difference between the esimtated performance given the model selection and the expected generalization performance (i.e., r=0). The metric used in this study is the Pearson correlation coefficient between the predicted and observed values. If one data sampling is considered as one iteration, 1000 iterations are conducted in this simulation to examine the distribution of the validation bias. A t-test is used to examine if the mean of the validation bias is significantly different from zero.

### Result

A diagram of a graph

Description automatically generated

The bias were visualized in box plots, where the factor $FS$ is presented on the x-axis, and $HT$ is colored in green and yellow to indicate the result of incocrectly and correctly implement the model selection, respectively. The y-axis is the validation bias by correlatino coefficent, with a horizontal line r=0 to indicate the expected generalization performance. An obvious overestimated performance is shown when the feaeture selection is carried out for the entire dataset with or without proper hyperparameter tuning. The median bias are 0.797 and 0.761 for $FS=0;HT=0$ and $FS=0;HT=1$, respectively. In addition, even when implementing the feature selection within each cross validation fold, falsely validate the hyperpamater will result a significant bias (p-value < 0.001) with a median of 0.113 ($FS=1; HT=0$). The only unbiased esimation is to include both feature selection and hyperparameter tuning in the cross validation process ($FS=1; HT=1$), where the median bias is -0.008. The result is consistent with the hypothesis and literature that the model selection should be included in the cross validation process to avoid overestimating the model performance.

### Suggestion

Use cross validation wrap-up function to do the cross validation in the inner loop.

The accuracy of the kernel machine on test data is critically dependent on the choice of good values for the hyper-parameters, in this case λ and

Block Cross Validation

In experimental design, blocking is an essentail approach to cotnrol for variation that can confound the primary effects of interest. For example, when investigating the impact of feed composition on the milk production of individual cow, day of lactation should be a blocking facotr (ref) since milk yield is known to change over the course of lactation, and the effect of feed composition may be confounded with the effect of lactation day. Since the variation of blocking factor may contribute to the trait of interest, it is essential to consider the blocking factor when evaluating the model performance. When the inter-block variation is larger than the examined source of interest (i.e., feed composition in the previous example), randomly splitting the data in cross validation may lead to biased estimation of the model performance. Since the model may achieve high accuracy by simply learning the variation of the blocking factor. In this case, a block cross validation, in which each experimental block is used as a test set, is recommended to avoid the overotpimistic conclusion.

Past studies ...

### Objectives and Hypothesis

The objective of the simulation study is to demonstrate how a regular cross validation (denoted as random CV), which randomly assign the samples to folds without considering the block effects, could overestimate the model performance. This study also conducts a block cross validation (denoted as block CV), where each block is used as a fold in the cross validation, as the benchmark. The hypothesis is that the model performance estimated by random CV is significantly higher than the estimation by block CV.

### Simulation Design

Similar to the study 1 and 2, a regression task is simulated which has 100 observations for ten predictors $X$ and one response variable ($Y$). Both $X$ and $Y$ are sampled from a standard normal distribution. The block factor is simulated by grouping 20 observations as a block, wich each block has an increasement of 3 units from zero. Among the ten predictors, one of which is replaced as the block levels, which is an integer value from 0 to 4, plus a random noise sampled from a standard normal distribution. This is to simulate a case where the modeled predictors can only capture the block variation, since the expected predicatiblity of using ten random variables ($X$) to predict another random variable (i.e., $Y$) is zero. Two model validation strategies, block CV and random CV, are examined in this study. Both strategies are 5-fold cross validation, in which block CV uses each block as a fold, and random CV randomly assign the samples to each fold (figure x1). The prediction model and the evaluation metric used are linear regression and Pearson's correlation coefficient, respectively. The simulation is repeated 1000 iterations, which $X$ and $Y$ are re-sampled in each iteration. A single tail t-test is used to examine if the mean of the estimated performance is significantly higher than zero. An analysis of variance (ANOVA) table is also computed to confirm if the simulated block variation is significantly larger than the residual variation.

### Result

The ANOVA table (table x) is calculated from one iteration as the demonstration, which show that the simulated data has the block variation significantly greater than the residual variance. A significant difference (p-value < 0.001) of the estimated performance is observed, with a mean of -0.001 and 0.768 for block CV and random CV, respectively. The result is consistent with the hypothesis that the random CV will overestimate the model performance when the block variation is larger than the residual variation.

A diagram of a cross-validation strategy

Description automatically generated